CENG463 Term Project

Hasan Basri Paşazade 21050151005 Mehmet Baran Geylani 21050151004

**Abstract**

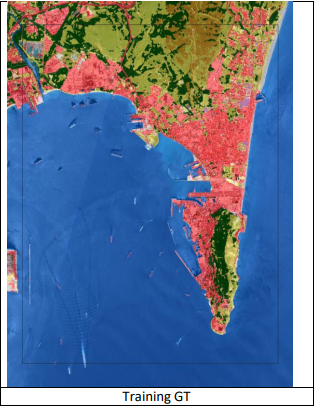
This document contains the detailed description of CENG463 Term Project problem and how our team approached to this problem, how the data was prepared, which algorithms were used to solve the problem and which ideas are behind the selection of those algorithms.

**CENG 463 MACHINE LEARNING TERM PROJECT REPORT**

**Github Link:** https://github.com/brngylni/CENG463\_Project

**Problem**

Climate change has become a serious threat to humanity. Global land cover maps serve as an important means to tackle this problem. ESA has produced Global Land Cover Map from Sentinel-2 data. We generated land cover maps for unseen data by using a limited number of training samples. Gibraltar is selected as a study area as there are wide variety of land cover types. The csv file which contains the limited number of samples is train.csv and contains 967004 rows. Each row has Id, Code, Blue, Green, Red and NIR labels. These labels refer to the colors which are used in train\_gt.tif file.

****

**Preparing Data**

We prepared train.csv training file by doing some operations:

* Color values (Blue, Green, Red,) and NIR is normalized by dividing by 10000 each one of them.
* We added some new values to the data to help training part of the model:
  + NDVI feature: NDVI quantifies vegetation by measuring the difference between near-infrared and red light.

NDVI = (NIR - Red) / (NIR + RED)

We understood that when we have negative NDVI values, it is generally water. When we have positive, it is land.

* + NDWI feature: NDWI is used to monitor changes related to water content in water bodies.

NDWI = (Green - NIR) /

(Green + NIR)

* + EVI feature: EVI is an 'optimized' vegetation index designed to enhance the vegetation signal

EVI = 2.5 \* ((NIR - Red) / (NIR + 2.4\*Red + 1))

* + Color Intensity of Tif = (Red \* 0.2126) + 0.7152 \*  (Green + 0.0722) \* (Blue)
  + Every counted code value:
    - 10 🡪 57999
    - 20 🡪 15703
    - 30 🡪 108240
    - 40 🡪 4342
    - 50 🡪 143382
    - 60 🡪 15014
    - 80 🡪 618857
    - 90 🡪 1745

**Training Data**

We tried several classification methods and measure f1\_score for every method while we train data:

1. KNN: It is easy to implement and understand. It can handle multi-class cases like this one. KNN algorithm is used for applications that require high accuracy but that do not require a human-readable model.

We measured f1\_score 0.89 when test size is 0.1 and k value is 9.

1. Decision Tree: It is a very good solver for classification problems. It is used to categorize objects against learned features. Easy to understand. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. We measured f1\_score 0.8647031577043182, when test size is 0.1.
2. Random Forest SVC: is suitable for situations when we have a large dataset, and interpretability is not a major concern. It is faster than SVM methods. We implemented this method with multiple thread programming. In this method we used 30 leaves and measured f1\_score as 0. 9013530038010781 with 6 leaf and 0.1 test size.

**Predicting**

We obtained the results and picked the method that has the most successful f1\_score: Random Forest. Then we predicted the test data that has approximately 4 million rows. Finally prepared the format for the submission.

Our best Kaggle score is 0.35008